Performance Characterization of the Tile Processor Architecture: Lessons Learned

Eric Grobelny, Jim Passwater, and Andrew White
Honeywell Aerospace, Defense, and Space Systems
Clearwater, FL
eric.grobelny@honeywell.com

Abstract

As silicon technologies progress, they get closer and closer to their physical limitations. No longer can we squeeze more performance from increasing clock frequencies, but rather, we must look to parallelization to improve performance. As feature sizes shrink and the number of transistors on a single die increases, it becomes advantageous to incorporate more than one compute core into the chip. In fact, most CPUs sold today have at least two cores. While the desktop and workstation realm of commercial computing typically max out at eight cores, high-performance computing platforms are using 10s to 100s of cores.

The Tile64 from Tilera is one such device that uses 64 general-purpose processing cores interconnected in a mesh topology to provide massive parallelism with high-performance peripherals to support memory and I/O operations off-chip [1]. Figure 1 illustrates the Tile64 architecture and its various components and interconnects.

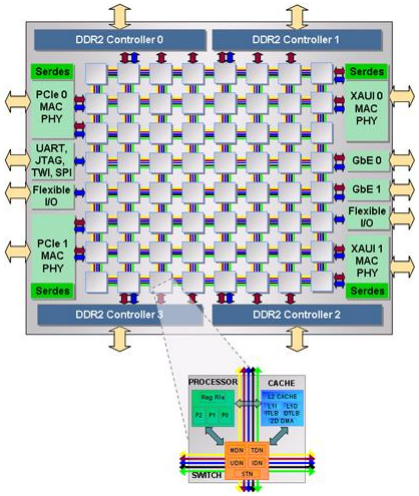

Figure 1: Tile64 architectural diagram [courtesy of Tilera].

While the capabilities and theoretical performance of the Tile64 are impressive on paper, efficiently utilizing its topology and resources are paramount in order to squeeze the most performance from the device. In fact, care must be taken to ensure highly efficient code when dealing with not only the architecture, but also the compiler [2]. In this paper, we will discuss four topic areas that deal with efficient use of the complex Tilera architecture. These topics include:

1) **Memory interfaces** – focusing on DMA writes/reads to and from various banks of memory with and without the use of shared memory
2) **Inter-tile networks** – focusing on the performance of the User Dynamic Network (UDN), the Memory Dynamic Network (MDN), and the I/O Dynamic Network (IDN).
3) **Inter-tile communication mechanisms** – focusing on message passing and channels (buffered, streaming, and raw channel types)
4) **Power-saving features/techniques** – focusing on sleep mode and other ways to reduce power consumption

For each of these topics, we will discuss the setup, benchmarks, and tests executed to evaluate the performance implications of using one mechanism/library/programming model over another. Rather than explaining the reasons behind our reported performance numbers, we will focus on the "lessons learned" by demonstrating the performance gains/losses under various scenarios and providing recommendations for using the best mechanisms for those scenarios.

After presenting the lessons learned from the Tile64, we will apply them to an innovative, low-cost development board called Symphony [3] that incorporates a Xilinx FPGA with a radiation-tolerant version of the Tile64 called Maestro. Maestro was developed under the OPERA program in order to bring high-performance computing to space application. For more details on Maestro and Opera, see [4]. The analysis conducted using Maestro will provide insight on the advantages and disadvantages of the radiation-tolerant chip over the vanilla Tile64. It will also be used to demonstrate the real-world gains realized through the optimizations identified in the previous study.

References

[1] Tilera Corporation, "Tile Processor Architecture Overview," Tilera Corporation, Doc. No. UG100, Release 1.1, April 2009.

[2] J. Richardson, C. Massie, H. Lam, K. Gosrani, and A. George, "Space Applications on Tilera," 3rd IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT), Pasadena, CA, July 2009.

[3] A. White, "The Honeywell Symphony Maestro Exploration Platform." Internal document, Honeywell, Space Electronic Systems, Clearwater, FL, 2010.*

[4] M. Malone, "OPERA RHBD Multi-core," Military and Aerospace Programmable Logic Devices (MAPLD) Conference, Greenbelt, MD, August 2009.

*Available upon request.